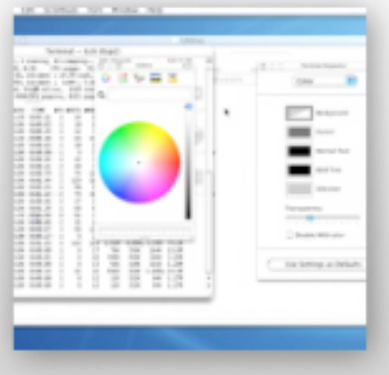
CFNetwork Overview

Session 112



CFNetwork Overview

Becky Willrich
Core Services Frameworks

CFNetwork Overview

- Overview of CFNetwork in the system
- Basics of using CFNetwork objects
- Basic abstractions
  - CFHost (new in Panther)
  - CFSocketStream
- Protocol implementations
  - CFFTPStream (new in Panther)
  - CFHTTPMessage and CFHTTPStream
  - CFNetServices (Rendezvous)

Overview

CFNetwork In Mac OS X

- What is CFNetwork?
- How does it fit in?
- When should you use it?

What Is CFNetwork?

- Library of abstractions for network protocols
  - DNS host resolution (new!)
  - FTP (new!)
  - HTTP
  - Rendezvous
  - Sockets
- APIs in the style of CoreFoundation
- APIs support asynchrony via the run loop

Goals of CFNetwork

- Expose the full power of the underlying protocol
- Strong integration with existing libraries and paradigms
  - Especially the run loop!
- High performance, with negligible overhead compared to raw sockets
- Will not protect you from the details!

How Does CFNetwork Fit In?

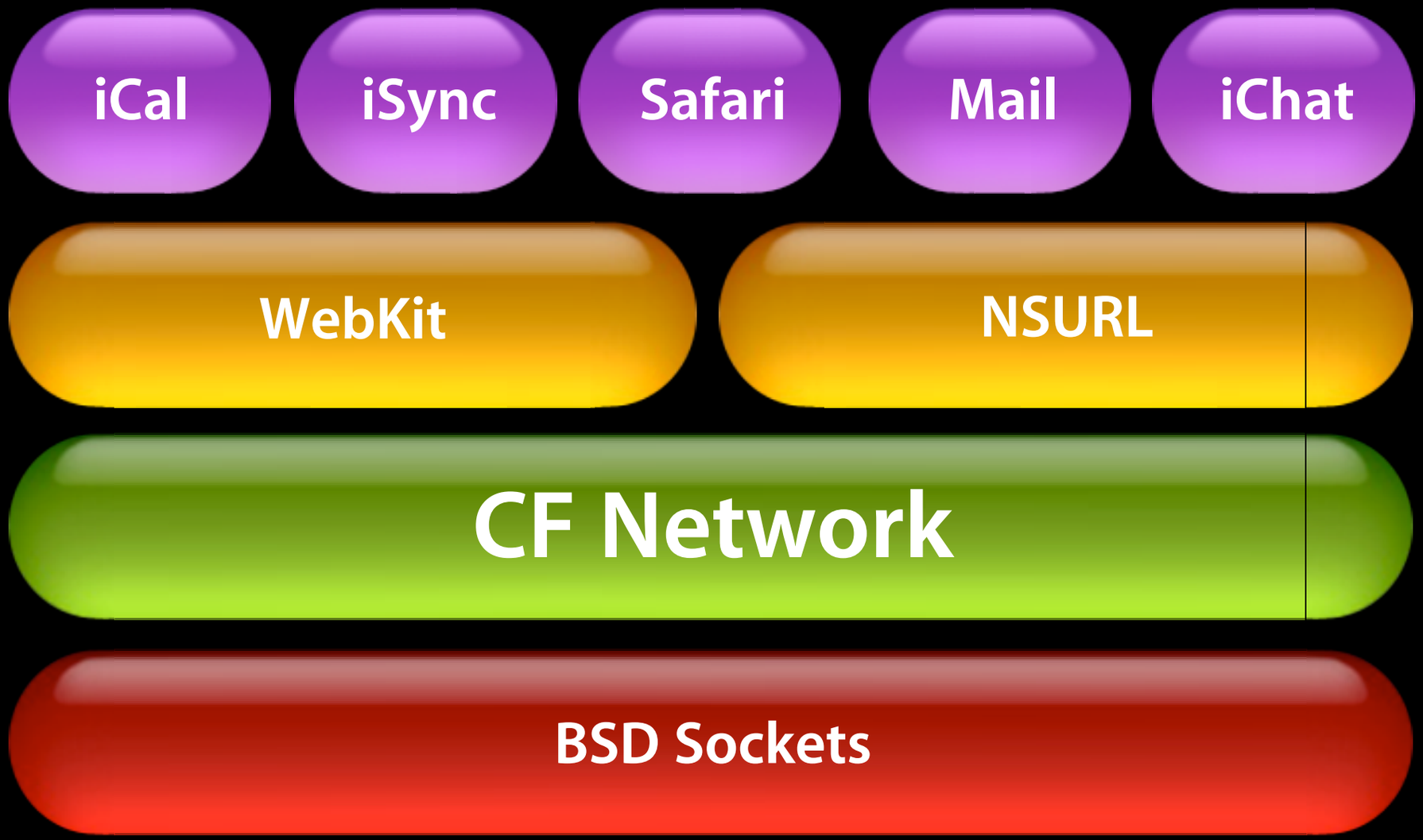
Applications

Application Frameworks

Core Services

Darwin

How Does CFNetwork Fit In?

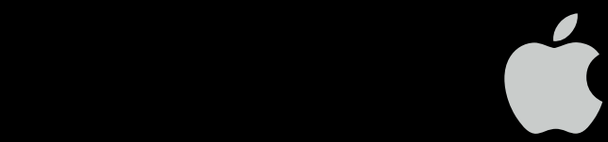


When Should You Use CFNetwork?

- Use CFNetwork if you need a low-level, high-performance framework
  - Need control of when bytes are read/written
  - Need detailed control of the protocol stack
  - Want to link low
- Use the higher APIs instead if possible

When Should You Use CFNetwork?

- Use CFNetwork if raw sockets are too low:
  - Want run loop integration
  - Want to avoid managing select() directly
  - Want simple TLS/SSL use
  - Want an “object” abstraction (reference counting)
- Use CFNetwork if you know the protocol, but don’t want to implement it



When Should You NOT Use CFNetwork?

- Use NSURLConnection instead if:
  - You want a convenience API
    (i.e., you want most choices made for you)
  - You must handle arbitrary URLs
  - You need to add your own protocol implementation
  - You want someone else to handle caching
  - You have sophisticated authentication needs

Basic Usage

Basic Usage Model

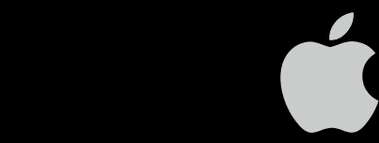
- Create the CFNetwork object you want to use
- Set yourself as the client of that object
- Schedule the object on the run loop
- Start the CFNetwork object
- Run the run loop (AppKit and HLTB do this for you)
- Wait for your callback to be called
- Clean up and dispose the object when you are done

All CFNetwork objects follow this model!

Example: CFHost

Create the CFNetwork object

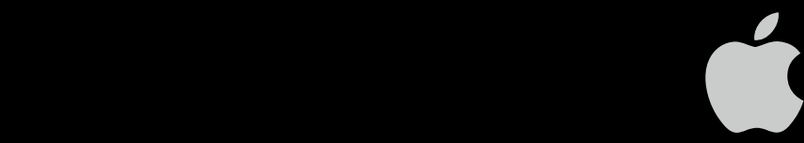
```
CFHostRef myHost =
   CFHostCreateWithName(NULL, CFSTR("www.apple.com"));

CFHostClientContext context =
   {0, myInfoPointer, NULL, NULL, NULL};
CFHostSetClient(myHost, myCallback, &context);

// This schedules us on the current thread
CFHostScheduleWithRunLoop(myHost, CFRunLoopGetCurrent(),
   kCFRunLoopModeDefault);

// Start IP address resolution
CFHostStartInfoResolution(myHost, kCFHostAddresses, NULL);

CFRunLoopRun();
```



Example: CFHost

Set the client

```
CFHostRef myHost =
   CFHostCreateWithName(NULL, CFSTR("www.apple.com"));

CFHostClientContext context =
   {0, myInfoPointer, NULL, NULL, NULL};
CFHostSetClient(myHost, myCallback, &context);

// This schedules us on the current thread
CFHostScheduleWithRunLoop(myHost, CFRunLoopGetCurrent(),
   kCFRunLoopModeDefault);

// Start IP address resolution
CFHostStartInfoResolution(myHost, kCFHostAddresses, NULL);

CFRunLoopRun();
```

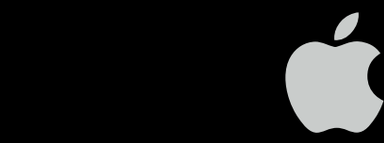


Example: CFHost

Schedule on the run loop

```
CFHostRef myHost =
   CFHostCreateWithName(NULL, CFSTR(”www.apple.com”));

CFHostClientContext context =
   {0, myInfoPointer, NULL, NULL, NULL};
CFHostSetClient(myHost, myCallback, &context);
```

```
// This schedules us on the current thread
CFHostScheduleWithRunLoop(myHost, CFRunLoopGetCurrent(),
   kCFRunLoopModeDefault);

// Start IP address resolution
CFHostStartInfoResolution(myHost, kCFHostAddresses, NULL);

CFRunLoopRun();
```



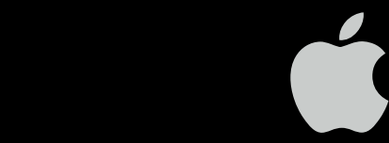
Example: CFHost

Start the object

```
CFHostRef myHost =
   CFHostCreateWithName(NULL, CFSTR(”www.apple.com”));

CFHostClientContext context =
   {0, myInfoPointer, NULL, NULL, NULL};
CFHostSetClient(myHost, myCallback, &context);
```

```
// This schedules us on the current thread
CFHostScheduleWithRunLoop(myHost, CFRunLoopGetCurrent(),
   kCFRunLoopModeDefault);
```

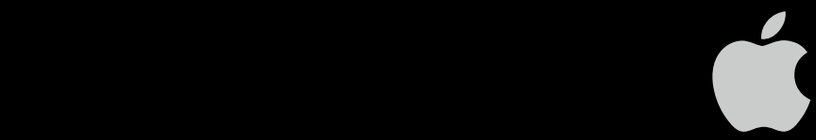
```
// Start IP address resolution
CFHostStartInfoResolution(myHost, kCFHostAddresses, NULL);

CFRunLoopRun();
```



Example: CFHost

Fielding the callback

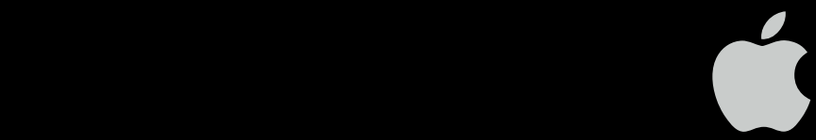
```
void myCallback(CFHostRef myHost, CFHostInfoType typeInfo,
const CFStreamError *error, void *myInfoPointer) {
    if (error && error->error != 0) {
        ... handle error ...
    } else if (typeInfo == kCFHostAddresses) {
        CFArrayRef addressArray =
            CFHostGetAddressing(myHost, NULL);
        // second arg is hasBeenResolved; we know this is true
        processAddresses(addressArray);
    }
}
```



Example: CFHost

Cleaning up

```
// Guarantees your callback will never be called again
CFHostSetClient(myHost, NULL, NULL);

// Prevents wasting CPU spinning the run loop
CFHostUnscheduleFromRunLoop(myHost, CFRunLoopGetCurrent(),
   kCFRunLoopModeDefault);

// If the CFNetwork object is still doing work, you should
// stop that work here by calling CFHostCancelInfoResolution(),
// CFReadStreamClose(), CFNetServiceCancel()...

// Finally, release your reference
CFRelease(myHost);
myHost = NULL;
```



Other Threading Models

- The basic model is event-driven
  - You are told when interesting things have happened
- Other models are supported
  (see the documentation for how)
  - Polling
  - Blocking
- **But**…
  - They're more difficult to use
  - They don't perform as well
  - We don't recommend them

Basic Abstractions

Basic Abstractions

- CFHost
- CFSocketStream

CFHost

- New in Panther!
- CFHost represents a remote DNS host
- Lets you resolve the host asynchronously
- Created from either a host name or from an IP address
  - `CFHostCreateWithName()`
  - `CFHostCreateWithAddress()`

CFHost

- Call `CFHostStartInfoResolution()` to start a resolution
  - Arguments passed determines what's resolved
  - Can resolve:
    - IP addresses
    - DNS names (reverse DNS lookup)
    - Reachability (as defined by SystemConfiguration.framework)
- When your callback is called, the information is available
- Call in to CFHost to retrieve it
  - `CFHostGetAddressing()`
  - `CFHostGetNames()`
  - `CFHostGetReachability()`

CFSocketStream

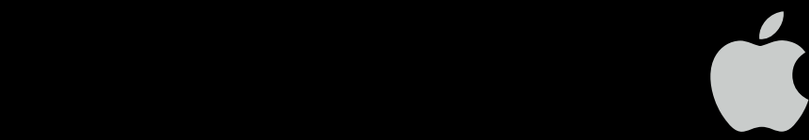
- Create a read stream and write stream pair for a socket
  - To a CFHost
  - To a CFNetService
  - To a hostname and port
  - To a socket signature
  - By raw socket
- Configuring one stream affects the other

CFSocketStream

- Basic support is in CoreFoundation/CFStream.h
- CFNetwork adds several configuration options
  - SSL/TLS encryption and authentication
  - SOCKS proxy support
  - Look in CFNetwork/CFSocketStream.h for more information

A Word About Proxies…

- Several CFNetwork APIs accept proxy dictionaries
  - CFSocketStream
  - CFHTTPStream
  - CFFTPStream
- Proxies are never applied automatically
- You can always pass the dictionary from SystemConfiguration.framework
  - `SCDynamicStoreCopyProxies()` returns the current proxy settings
- You can also build your own for custom configuration



Network Protocols

Network Protocols

- FTP
  - CFFTPStream
- HTTP
  - CFHTTPMessage
  - CFHTTPStream
- Rendezvous
  - CFNetService
  - CFNetServiceBrowser

CFFTPStream

- New in Panther
- Use CFFTPStream to:
  - Download an FTP URL
  - Download an FTP directory listing
  - Upload coming in Panther

Downloading an FTP URL

```
// Create the URL
CFURLRef url = CFURLCreateWithString(NULL,
  CFSTR(”ftp://ftp.myServer.com/myFile”), NULL);

// Create the stream
CFReadStreamRef stream =
  CFReadStreamCreateWithFTPURL(NULL, url);

// Now use it like any other stream...

CFReadStreamOpen(stream);
while (CFReadStreamGetStatus(stream)!=kCFStreamStatusAtEnd)
{
  CFReadStreamRead(stream, buf, bufSize);
  ....
}
```

Downloading a Directory Listing

```
// Make sure the URL is a directory URL
CFURLRef url = CFURLCreateWithString(NULL,
  CFSTR(”ftp://ftp.apple.com/developer/”), NULL);

CFReadStreamRef stream =
  CFReadStreamCreateWithFTPURL(NULL, url);

/* As bytes are read, we pass them to
CFFTPCreateParsedResourceListing to get the listing */

CFReadStreamOpen(stream);
while (CFReadStreamGetStatus(stream)!=kCFStreamStatusAtEnd)
{
  CFReadStreamRead(stream, buf, bufSize);
  parseResourceListing(buf, bufSize);
}
```

Downloading a Directory Listing

```
void parseResourceListing(UInt8 *buf, int bufSize) {
  CFIndex bytesConsumed, startIndex = 0;
  CFDictionaryRef newListing;

do {
  bytesConsumed = CFFTPCreateParsedResourceListing(
    NULL,                 // allocator
    buf + startIndex,     // bytes to parse
    bufSize - startIndex, // num bytes to parse
    &newListing);         // the parsed listing

  listingName =
    CFDictionaryGetValue(newListing, kCFFTPResourceName);
  CFShow(listingName);

  startIndex += bytesConsumed;
} while (bytesConsumed > 0);
// 0 means insufficient bytes; -1 means parse error
```

CFHTTP

- CFHTTPMessage represents an HTTP request or response
- CFHTTPStream performs an HTTP transaction
- Normal usage:
  - Create a CFHTTPMessage for a request
  - Create a CFHTTPStream from the message
  - Open and read from the stream to process the request

CFHTTP

```
// Create the URL
CFURLRef url = CFURLCreateWithString(NULL,
  CFSTR(”http://www.apple.com/”), NULL);

// Create the HTTP request
CFHTTPMessageRef request = CFHTTPMessageCreateRequest(NULL,
  CFSTR(”GET”), url, kCFHTTPVersion1_1);

// Configure the request
CFHTTPMessageSetHeaderFieldValue(request,
  CFSTR(”User-Agent”), CFSTR(”MyApp/1.0”));

// Create the stream
stream = CFReadStreamCreateForHTTPRequest(NULL, request);

// Now use it like any other stream...
```

Other Usages

- CFHTTPMessage can be used to parse an HTTP header
  - `CFHTTPMessageCreateEmpty()`, `CFHTTPMessageAppendBytes()`
- CFHTTPStream can stream large request bodies
  - `CFReadStreamCreateForStreamedHTTPRequest()`
- CFHTTPMessage can add authentication information
  - `CFHTTPMessageAddAuthentication()`

Rendezvous

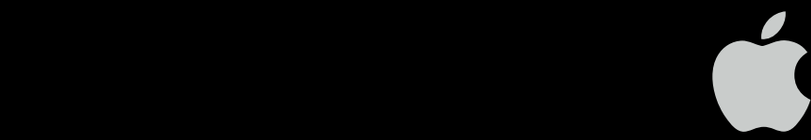
- Introduced in Jaguar
- Allows clients to advertise their services
- Those services can then be discovered by other clients
- CFNetwork integrates these services with the run loop

CFNetServices

- Network Service (CFNetService)
  - Represents a single service provider on the network
  - Attributes are domain, type, name, address, and port
  - Use it:
    - To advertise a service you provide
    - To get the details of someone else's service
- Network Service Browser (CFNetServiceBrowser)
  - Represents a search
    - For services
    - For service domains

Using CFNetService

- Advertising your service
  - Create your service via `CFNetServiceCreate()`
  - Register using `CFNetServiceRegister()`
  - Your callback is only called if/when a name conflict occurs
- Resolving someone else's service
  - You get a CFNetService from a CFNetServiceBrowser
  - Pass it directly to `CFStreamCreatePairWithSocketToNetService()` to connect immediately
  - Or call `CFNetServiceResolve()` to get the details about the net service yourself



Using CFNetServiceBrowser

- Create the browser with `CFNetServiceBrowserCreate()`
- Perform the search you're interested in
  - `CFNetServiceBrowserSearchForDomains()` looks for service domains
  - `CFNetServiceBrowserSearchForServices()` looks for services in a given domain
- Your callback is called as domains/services are found

Wrap Up

Roadmap

403 Internet Technologies: Safari Overview
Discusses the pieces that go in to Safari

Presidio
Tue., 2:00pm

105 Rendezvous
Zero Configuration Networking with Rendezvous

Marina
Wed., 5:00pm

110 Networking: Overview of Mac OS X Networking APIs
Discusses all the networking APIs and what's new

Mission
Thur., 10:30am

418 Internet Technologies: Advanced Foundation URL APIs
The next layer up from CFNetwork

Nob Hill
Thur., 3:30pm

Who to Contact

Xavier Legros
Mac OS X Evangelist
xavier@apple.com

http://developer.apple.com/wwdc2003/urls.html

For More Information

Documentation

http://developer.apple.com/documentation/

CFNetwork

ADC Home > Documentation > Networking >
http://developer.apple.com/documentation/Networking/Conceptual/CFNetwork

CFRunLoop

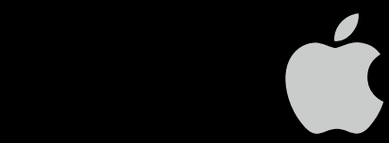
http://developer.apple.com/documentation/CoreFoundation/Reference/CFRunLoopRef

Rendezvous

http://developer.apple.com/macosx/rendezvous

Sample Code
/Developer/Examples/Networking

- Echo
  - Implements an echo server
  - Advertises itself via Rendezvous
- Echo Client
  - Implements an echo client
  - Discovers servers via Rendezvous
- GET Example, POST Example
  - Uses CFHTTPMessage, CFHTTPStream to perform a GET and POST
- URLLoad
  - Loads arbitrary URLs via several different APIs

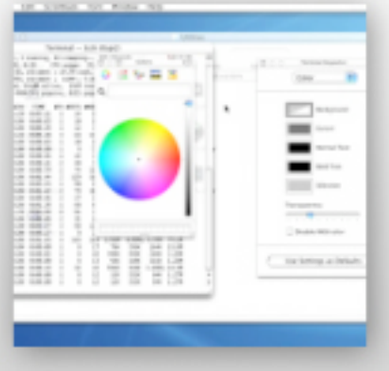


Mailing List

macnetworkprog@lists.apple.com

- Mailing list for discussing all aspects of network programming on MacOS
- Not for system administrators
- Engineers from all the networking teams at Apple are on the list
- Subscribe at http://lists.apple.com/mailman/listinfo/macnetworkprog
- Or follow the links at http://lists.apple.com/

Q&A



Xavier Legros
Mac OS X Evangelist, Favorite French Guy
xavier@apple.com

http://developer.apple.com/wwdc2003/urls.html

WWDC 2003

Worldwide Developers Conference